

**WE CLAIM:**

1 1. A computer system providing type support for multiple type  
2 definitions, comprising:

3 an interface repository including:

4 a repository naming context; and,

5 a prefix naming context subordinate to the repository  
6 naming context, the prefix naming context serving as a root  
7 naming context for at least one interface definition language  
8 declaration.

1 2. The system of claim 1 wherein the prefix naming context further  
2 includes:

3 at least one naming context defined by an interface definition  
4 object and subordinate to the prefix naming context.

1 3. The computer system of claim 2 wherein at least one interface  
2 definition object has a fully scoped object name including a prefix name of the  
3 prefix naming context to which the interface definition object is subordinated.

1 4. The computer system of claim 1 wherein the prefix naming context is  
2 immediately subordinate to the repository naming context.

1 5. The computer system of claim 1 wherein the prefix naming context  
2 further includes:

3           at least one leaf node defined by an interface definition object.

1           6. The computer system of claim 1, wherein the prefix naming context is  
2 defined by a prefix object.

1           7. The computer system of claim 1, further comprising:

2                 an interface repository loader that accepts as input parameters a  
3 specified interface definition language file containing at least one  
4 interface definition language declaration, and a specified prefix name,  
5 and installs the at least one interface definition language declaration in  
6 a prefix naming context having the prefix naming context in the  
7 interface repository.

1           8. The computer system of claim 7, wherein the interface repository  
2 loader creates a data file identified as related to the specified interface  
3 definition language file, and containing an identification of the specified  
4 prefix naming context.

1           9. The computer system of claim 7, wherein the interface repository  
2 loader creates the specified prefix naming context in the interface repository if  
3 the specified prefix naming context does not exist therein.

1           10. The computer system of claim 1, further comprising:

2                 a memory device that stores the interface repository; and

3                 a processing unit that executes operations of the interface  
4 repository loader.

1 11. The computer system of claim 10, wherein the processing unit further  
2 executes the interface repository loader to create a data file identified as related  
3 to the specified interface definition language file, and containing an  
4 identification of the specified prefix naming context.

1 12. A computer system providing type support for multiple type  
2 definitions, comprising:

3 at least one client object having a stub routine including a fully  
4 scoped name identifying a type providing an operation to the client  
5 object; and,

6 at least one server object having a skeleton routine including a  
7 fully scoped name identifying a type for the server object.

1 13. The computer system of claim 12, further comprising:

2 an interface definition language compiler that generates the stub  
3 routine in a client object.

1 14. The computer system of claim 12, further comprising:

2 an interface definition language compiler that generates a  
3 skeleton routine in a server object.

1 15. The computer system of claim 12, further comprising:

2 a memory device that stores the at least one client object, and the  
3 at least one server object; and,

4 a processing unit that executes a server object in response to an  
5 invocation of the server object by a client object.

1 16. The computer system of claim 15, wherein the processing unit executes  
2 an interface definition language compiler to generate the stub routine and the  
3 skeleton routine.

1 17. A method of providing type support for multiple type definitions,  
2 comprising the steps of:

3 defining in an interface repository a prefix naming context; and

4 storing the prefix naming context subordinate to a repository  
5 naming context in the interface repository, the prefix naming context  
6 forming an interface definition language root context for interface  
7 definition objects subordinate to the prefix naming context.

1 18. The method of claim 17, wherein each prefix naming context is stored  
2 immediately subordinate to the repository naming context.

1 19. The method of claim 17 further comprising the steps of:

2 specifying an interface definition language file containing at least  
3 one interface definition language declaration;

4 specifying a prefix naming context; and

5 storing each interface definition language declaration in the  
6 specified interface definition language file into the specified prefix  
7 naming context.

1        20. The method of claim 19, wherein the step of storing each interface  
2 definition language declaration further comprises the steps of:

3                creating an interface definition object for the interface definition  
4 language declaration;

5                storing the interface definition object in the specified prefix  
6 naming context; and

7                providing the interface definition object with a fully scoped  
8 object name including a prefix name from the prefix naming context in  
9 which the interface definition object is stored.

1        21. The method of claim 19, further comprising the step of:

2                creating a data file identified as related to the specified interface  
3 definition language file, and containing an identification of the  
4 specified prefix naming context.

1        22. A method of providing type support for multiple type definitions,  
2 comprising the step of:

3                providing an interface repository including:

4                        a repository naming context; and

5                        a prefix naming context subordinate to the repository  
6 naming context, the prefix naming context serving as a root  
7 naming context for at least one interface definition language  
8 declaration.

1 23. A method of providing type support for multiple type definitions,  
2 comprising the steps of

3 providing at least one client object having a stub routine  
4 including a fully scoped name for an object type providing an  
5 operation to the client object; and

6 providing at least one server object having a skeleton routine  
7 including a fully scoped name identifying an object type for the server  
8 object.

1 24. The method of claim 21, further comprising the step of:

2 providing a memory device that stores the at least one client  
3 object, and the at least one server object; and

4 providing a processing unit that executes a server object in  
5 response to an invocation of the server object by a client object.

add A2

add  
B2